(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号

特開平9－288580

(43)公開日　平成９年(1997)11月４日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　9/45			Ｇ０６Ｆ　9/44	３２２Ｆ	

審査請求　有　　請求項の数14　ＯＬ　（全 13 頁）

(21)出願番号　特願平8－98045

(22)出願日　平成８年(1996)４月19日

(71)出願人　000004237
日本電気株式会社
東京都港区芝五丁目７番１号

(72)発明者　田中　精一
東京都港区芝五丁目７番１号　日本電気株式会社内

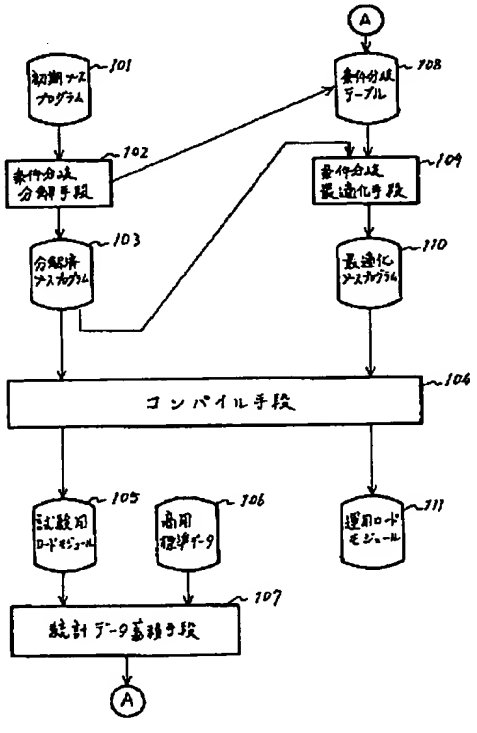
(74)代理人　弁理士　京本　直樹　（外２名）

(54)【発明の名称】　ソースプログラムの最適化装置および最適化方法

(57)【要約】

【課題】複雑な多段階命令を含むソースプログラムを最適化して高速に実行可能なロードモジュールを作成する。

【解決手段】条件分岐分解手段が、ソースプログラムの条件分岐命令を解析するとともに、ソースプログラムの条件分岐命令内の各条件毎の処理とその実行件数との対応情報を含む条件分岐テーブルを作成し、統計データ蓄積手段が、ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測して実行件数を条件分岐テーブルに記憶し、条件分岐最適化手段が、条件分岐テーブル内の各処理の実行件数に基づいて、ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する。



1

【特許請求の範囲】

【請求項１】　ソースプログラムの条件分岐命令内の各条件毎の処理を実行頻度の順に並べ替えることにより、ソースプログラムを最適化する条件分岐最適化手段を備えることを特徴とするソースプログラムの最適化装置。

【請求項２】　ソースプログラムの条件分岐命令を解析し、該ソースプログラムの条件分岐命令内の各条件毎の処理とその実行件数との対応情報を含む条件分岐テーブルを作成する条件分岐分解手段と、

前記ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測し、該実行件数を前記条件分岐テーブルに記憶する統計データ蓄積手段と、

前記条件分岐テーブル内の各処理の実行件数に基づいて、前記ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する条件分岐最適化手段とを備えることを特徴とするソースプログラムの最適化装置。

【請求項３】　ソースプログラムの条件分岐命令を解析し、該ソースプログラムの条件分岐命令内の各条件毎の処理とその処理を行う条件とその実行件数との対応情報を含む条件分岐テーブルを作成する条件分岐分解手段と、

前記ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測し、該実行件数を前記条件分岐テーブルに記憶する統計データ蓄積手段と、

前記条件分岐テーブル内の各処理を行う条件に基づいて前記ソースプログラムにおける条件分岐命令を多肢選択型の条件分岐命令に変換するとともに、前記条件分岐テーブル内の各処理の実行件数に基づいて前記ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する条件分岐最適化手段とを備えることを特徴とするソースプログラムの最適化装置。

【請求項４】　前記条件分岐分解手段は、初期ソースプログラムの二者択一型の条件分岐文の直下に選択肢の一方が存在しない場合に、その位置にダミー処理を挿入するとともにチェックポイントを設定して分解済ソースプログラムを作成した後、該ダミー処理を前記条件分岐テーブルに登録し、

前記統計データ蓄積手段は、前記分解済ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行する際に、該試験用データが前記チェックポイントを通過する件数により各処理の実行件数を計測し、該実行件数を前記条件分岐テーブルに記憶することを特徴とする請求項２記載のソースプログラムの最適化装置。

【請求項５】　前記条件分岐分解手段は、さらに、初期ソースプログラムの二者択一型の条件分岐文の直下に選択肢の一方が存在する場合に、自身の処理を再帰的に呼び出すことを特徴とする請求項４記載のソースプログラムの最適化装置。

2

【請求項６】　前記条件分岐分解手段は、さらに、選択肢の直下に二者択一型の条件分岐文が存在しない場合に、その位置にチェックポイントを設定して分解済ソースプログラムを作成した後、その位置にある処理を前記条件分岐テーブルに登録することを特徴とする請求項５記載のソースプログラムの最適化装置。

【請求項７】　前記条件分岐最適化手段は、初期ソースプログラムの二者択一型の条件分岐文を多肢選択型の条件分岐文に変換した後、前記条件分岐テーブルにおける実行件数の多い順に対応する条件の記述を行い、さらに、その条件が満たされる場合に行うべき処理を記述することにより、最適化ソースプログラムを作成することを特徴とする請求項６記載のソースプログラムの最適化装置。

【請求項８】　ソースプログラムの条件分岐命令内の各条件毎の処理を実行頻度の順に並べ替えることにより、ソースプログラムを最適化するステップを含むことを特徴とするソースプログラムの最適化方法。

【請求項９】　ソースプログラムの条件分岐命令を解析し、該ソースプログラムの条件分岐命令内の各条件毎の処理とその実行件数との対応情報を記憶する第１のステップと、

前記ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測して記憶する第２のステップと、

前記第２のステップで記憶した実行件数に基づいて、前記ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する第３のステップとを含むことを特徴とするソースプログラムの最適化方法。

【請求項１０】　ソースプログラムの条件分岐命令を解析し、該ソースプログラムの条件分岐命令内の各条件毎の処理とその処理を行う条件とその実行件数との対応情報を記憶する第１のステップと、

前記ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測して記憶する第２のステップと、

前記第１のステップで記憶した条件に基づいて前記ソースプログラムにおける条件分岐命令を多肢選択型の条件分岐命令に変換するとともに、前記第２のステップで記憶した実行件数に基づいて前記ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する第３のステップとを含むことを特徴とするソースプログラムの最適化方法。

【請求項１１】　前記第１のステップで、初期ソースプログラムの二者択一型の条件分岐文の直下に選択肢の一方が存在しない場合に、その位置にダミー処理を挿入するとともにチェックポイントを設定して分解済ソースプログラムを作成した後、該ダミー処理を記憶し、
前記第２のステップで、前記分解済ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行する際に、該試験用データが前記チェックポイントを通過する件数により各処理の実行件数を計測して記憶することを特徴とする請求項１０記載のソースプログラムの最適化方法。

【請求項１２】　前記第１のステップで、さらに、初期ソースプログラムの二者択一型の条件分岐文の直下に選択肢の一方が存在する場合に、自身の処理を再帰的に呼び出すことを特徴とする請求項１１記載のソースプログラムの最適化方法。

【請求項１３】　前記第１のステップで、さらに、選択肢の直下に二者択一型の条件分岐文が存在しない場合に、その位置にチェックポイントを設定して分解済ソースプログラムを作成した後、その位置にある処理を記憶することを特徴とする請求項１２記載のソースプログラムの最適化方法。

【請求項１４】　前記第３のステップで、初期ソースプログラムの二者択一型の条件分岐文を多肢選択型の条件分岐文に変換した後、前記条件分岐テーブルにおける実行件数の多い順に対応する条件の記述を行い、さらに、その条件が満たされる場合に行うべき処理を記述することにより、最適化ソースプログラムを作成することを特徴とする請求項１３記載のソースプログラムの最適化方法。

【発明の詳細な説明】

【発明の属する技術分野】本発明は、ソースプログラムを最適化し、高速に実行可能なロードモジュールを生成するソースプログラムの最適化装置および最適化方法に関する。

【０００１】

【従来の技術】条件分岐命令を含むプログラムを高速に実行するため、そのプログラムの実行時に、条件分岐命令を構成する複数の条件節をその成立頻度の高い順に並べ替える方法が、特開平２－２４２５号公報に記載されている。この方法では、ＢＡＳＩＣ言語において、複数のＩＦ文からなる多方向分岐命令の高速化について、記載されている。

【０００２】

【発明が解決しようとする課題】しかし、この従来の技術においては、ＩＦ文からなる多方向分岐命令については検討されているが、複数のＩＦ文がいわゆる入れ子の構造をなす多段階分岐命令については考慮されていない。したがって、複雑な多段階分岐命令に対しては、高速化できないという問題点があった。

【０００３】また、ＢＡＳＩＣ言語を前提としていることからプログラムの実行時に条件節の並べ替えを行っており、高速化されたプログラムの内容をソースレベルでユーザに提供することができないという問題点もあった。

【０００４】本発明の目的は、複雑な多段階命令を含むソースプログラムを最適化して高速に実行可能なロードモジュールを作成することにある。

【０００５】本発明の他の目的は、最適化されたソースプログラムの記述内容をユーザに提供できるようにすることにある。

【０００６】

【課題を解決するための手段】本発明の第１のソースプログラムの最適化装置は、ソースプログラムの条件分岐命令内の各条件毎の処理を実行頻度の順に並べ替えることにより、ソースプログラムを最適化する条件分岐最適化手段から構成されている。

【０００７】本発明の第２のソースプログラムの最適化装置は、ソースプログラムの条件分岐命令を解析し、該ソースプログラムの条件分岐命令内の各条件毎の処理とその実行件数との対応情報を含む条件分岐テーブルを作成する条件分岐分解手段と、前記ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測し、該実行件数を前記条件分岐テーブルに記憶する統計データ蓄積手段と、前記条件分岐テーブル内の各処理の実行件数に基づいて、前記ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する条件分岐最適化手段とから構成されている。

【０００８】本発明の第３のソースプログラムの最適化装置は、ソースプログラムの条件分岐命令を解析し、該ソースプログラムの条件分岐命令内の各条件毎の処理とその処理を行う条件とその実行件数との対応情報を含む条件分岐テーブルを作成する条件分岐分解手段と、前記ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測し、該実行件数を前記条件分岐テーブルに記憶する統計データ蓄積手段と、前記条件分岐テーブル内の各処理を行う条件に基づいて前記ソースプログラムにおける条件分岐命令を多肢選択型の条件分岐命令に変換するとともに、前記条件分岐テーブル内の各処理の実行件数に基づいて前記ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する条件分岐最適化手段とから構成されている。

【０００９】本発明の第４のソースプログラムの最適化装置は、第２のソースプログラムの最適化装置において、前記条件分岐分解手段は、初期ソースプログラムの二者択一型の条件分岐文の直下に選択肢の一方が存在し

ない場合に、その位置にダミー処理を挿入するとともにチェックポイントを設定して分解済ソースプログラムを作成した後、該ダミー処理を前記条件分岐テーブルに登録し、前記統計データ蓄積手段は、前記分解済ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行する際に、該試験用データが前記チェックポイントを通過する件数により各処理の実行件数を計測し、該実行件数を前記条件分岐テーブルに記憶することを特徴とする。

【００１０】本発明の第５のソースプログラムの最適化装置は、第４のソースプログラムの最適化装置において、前記条件分岐分解手段は、さらに、初期ソースプログラムの二者択一型の条件分岐文の直下に選択肢の一方が存在する場合に、自身の処理を再帰的に呼び出すことを特徴とする。

【００１１】本発明の第６のソースプログラムの最適化装置は、第５のソースプログラムの最適化装置において、前記条件分岐分解手段は、さらに、選択肢の直下に二者択一型の条件分岐文が存在しない場合に、その位置にチェックポイントを設定して分解済ソースプログラムを作成した後、その位置にある処理を前記条件分岐テーブルに登録することを特徴とする。

【００１２】本発明の第７のソースプログラムの最適化装置は、第６のソースプログラムの最適化装置において、前記条件分岐最適化手段は、初期ソースプログラムの二者択一型の条件分岐文を多肢選択型の条件分岐文に変換した後、前記条件分岐テーブルにおける実行件数の多い順に対応する条件の記述を行い、さらに、その条件が満たされる場合に行うべき処理を記述することにより、最適化ソースプログラムを作成することを特徴とする。

【００１３】本発明の第１のソースプログラムの最適化方法は、ソースプログラムの条件分岐命令内の各条件毎の処理を実行頻度の順に並べ替えることにより、ソースプログラムを最適化するステップを含んでいる。

【００１４】本発明の第２のソースプログラムの最適化方法は、ソースプログラムの条件分岐命令を解析し、該ソースプログラムの条件分岐命令内の各条件毎の処理とその実行件数との対応情報を記憶する第１のステップと、前記ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測して記憶する第２のステップと、前記第２のステップで記憶した実行件数に基づいて、前記ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する第３のステップとを含んでいる。

【００１５】本発明の第３のソースプログラムの最適化方法は、ソースプログラムの条件分岐命令を解析し、該ソースプログラムの条件分岐命令内の各条件毎の処理と

その処理を行う条件とその実行件数との対応情報を記憶する第１のステップと、前記ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測して記憶する第２のステップと、前記第１のステップで記憶した条件に基づいて前記ソースプログラムにおける条件分岐命令を多肢選択型の条件分岐命令に変換するとともに、前記第２のステップで記憶した実行件数に基づいて前記ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する第３のステップとを含んでいる。

【００１６】本発明の第４のソースプログラムの最適化方法は、第３のソースプログラムの最適化方法において、前記第１のステップで、初期ソースプログラムの二者択一型の条件分岐文の直下に選択肢の一方が存在しない場合に、その位置にダミー処理を挿入するとともにチェックポイントを設定して分解済ソースプログラムを作成した後、該ダミー処理を記憶し、前記第２のステップで、前記分解済ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行する際に、該試験用データが前記チェックポイントを通過する件数により各処理の実行件数を計測して記憶することを特徴とする。

【００１７】本発明の第５のソースプログラムの最適化方法は、第４のソースプログラムの最適化方法において、前記第１のステップで、さらに、初期ソースプログラムの二者択一型の条件分岐文の直下に選択肢の一方が存在する場合に、自身の処理を再帰的に呼び出すことを特徴とする。

【００１８】本発明の第６のソースプログラムの最適化方法は、第５のソースプログラムの最適化方法において、前記第１のステップで、さらに、選択肢の直下に二者択一型の条件分岐文が存在しない場合に、その位置にチェックポイントを設定して分解済ソースプログラムを作成した後、その位置にある処理を記憶することを特徴とする。

【００１９】本発明の第７のソースプログラムの最適化方法は、第６のソースプログラムの最適化方法において、前記第３のステップで、初期ソースプログラムの二者択一型の条件分岐文を多肢選択型の条件分岐文に変換した後、前記条件分岐テーブルにおける実行件数の多い順に対応する条件の記述を行い、さらに、その条件が満たされる場合に行うべき処理を記述することにより、最適化ソースプログラムを作成することを特徴とする。

【００２０】本発明の第１のコンピュータにおいて読み取り可能な媒体は、ソースプログラムの条件分岐命令内の各条件毎の処理を実行頻度の順に並べ替えることにより、ソースプログラムを最適化するステップを含むプログラムを格納する。

【００２１】本発明の第２のコンピュータにおいて読み

取り可能な媒体は、ソースプログラムの条件分岐命令を解析し、該ソースプログラムの条件分岐命令内の各条件毎の処理とその実行件数との対応情報を記憶する第１のステップと、前記ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測して記憶する第２のステップと、前記第２のステップで記憶した実行件数に基づいて、前記ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する第３のステップとを含むプログラムを格納する。

【００２２】本発明の第３のコンピュータにおいて読み取り可能な媒体は、ソースプログラムの条件分岐命令を解析し、該ソースプログラムの条件分岐命令内の各条件毎の処理とその処理を行う条件とその実行件数との対応情報を記憶する第１のステップと、前記ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行することにより各処理の実行件数を計測して記憶する第２のステップと、前記第１のステップで記憶した条件に基づいて前記ソースプログラムにおける条件分岐命令を多肢選択型の条件分岐命令に変換するとともに、前記第２のステップで記憶した実行件数に基づいて前記ソースプログラムにおける条件分岐命令内の各条件毎の処理を並べ替えることにより、ソースプログラムを最適化する第３のステップとを含むプログラムを格納する。

【００２３】本発明の第４のコンピュータにおいて読み取り可能な媒体は、第３のコンピュータにおいて読み取り可能な媒体に格納されたプログラムにおいて、前記第１のステップで、初期ソースプログラムの二者択一型の条件分岐文の直下に選択肢の一方が存在しない場合に、その位置にダミー処理を挿入するとともにチェックポイントを設定して分解済ソースプログラムを作成した後、該ダミー処理を記憶し、前記第２のステップで、前記分解済ソースプログラムをコンパイルして作成された試験用ロードモジュールを試験用データを用いて実行する際に、該試験用データが前記チェックポイントを通過する件数により各処理の実行件数を計測して記憶することを特徴とする。

【００２４】本発明の第５のコンピュータにおいて読み取り可能な媒体は、第４のコンピュータにおいて読み取り可能な媒体に格納されたプログラムにおいて、前記第１のステップで、さらに、初期ソースプログラムの二者択一型の条件分岐文の直下に選択肢の一方が存在する場合に、自身の処理を再帰的に呼び出すことを特徴とする。

【００２５】本発明の第６のコンピュータにおいて読み取り可能な媒体は、第５のコンピュータにおいて読み取り可能な媒体に格納されたプログラムにおいて、前記第１のステップで、さらに、選択肢の直下に二者択一型の

条件分岐文が存在しない場合に、その位置にチェックポイントを設定して分解済ソースプログラムを作成した後、その位置にある処理を記憶することを特徴とする。

【００２６】本発明の第７のコンピュータにおいて読み取り可能な媒体は、第６のコンピュータにおいて読み取り可能な媒体に格納されたプログラムにおいて、前記第３のステップで、初期ソースプログラムの二者択一型の条件分岐文を多肢選択型の条件分岐文に変換した後、前記条件分岐テーブルにおける実行件数の多い順に対応する条件の記述を行い、さらに、その条件が満たされる場合に行うべき処理を記述することにより、最適化ソースプログラムを作成することを特徴とする。

【００２７】

【発明の実施の形態】次に、本発明の一実施例について図面を参照して詳細に説明する。

【００２８】図１を参照すると、本発明の一実施例であるソースプログラムの最適化装置、あるいは最適化方法を適用したシステムは、初期ソースプログラム１０１の条件分岐命令を解析して分解済ソースプログラム１０３を作成するとともに、条件分岐命令内の各条件毎の処理とその実行件数との対応情報を含む条件分岐テーブル１０８を作成する条件分岐分解手段１０２と、分解済ソースプログラム１０３をコンパイルして作成された試験用ロードモジュール１０５を商用標準データ１０６を用いて実行することにより各処理の実行件数を計測し、実行件数を条件分岐テーブル１０８に記憶する統計データ蓄積手段１０７と、条件分岐テーブル１０８内の各処理の実行件数に基づいて、分解済ソースプログラム１０３における条件分岐命令内の各条件毎の処理を並べ替えることにより、最適化ソースプログラム１１０を作成する条件分岐最適化手段１０９とを備えている。

【００２９】次に、本発明の一実施例であるソースプログラムの最適化装置、あるいは最適化方法を適用したシステムの動作について、図１～図７により説明する。

【００３０】図１を参照すると、条件分岐分解手段１０２は、初期ソースプログラム１０１を分解し、分解済ソースプログラム１０３を作成するとともに、条件分岐テーブル１０８に条件分岐データを登録する。条件分岐分解手段１０２の動作について、図２～図３を参照して説明する。

【００３１】図２に、初期ソースプログラム１０１、分解済ソースプログラム１０３、条件分岐テーブル１０８の一例を示す。

【００３２】この例では、初期ソースプログラム１０１は、条件Ａ、Ｂ、Ｃを有しており、条件Ａが真かつ条件Ｂが真の時に処理Ｐを、条件Ａが真かつ条件Ｂが偽かつ条件Ｃが偽の時に処理Ｑを実行するように組まれている。

【００３３】図３は、条件分岐分解手段１０２が初期ソースプログラム１０１を分解して分解済ソースプログラ

ム１０３を作成する際の動作をフローチャートにより示したものである。

【００３４】条件分岐分解手段１０２は、最適化の対象である初期ソースプログラム１０１を読み込み、ＩＦ文を検索することにより条件分岐命令であるか否かを判断する（ステップ３０１）。

【００３５】条件分岐命令である場合は、その下にＴＨＥＮ文が存在するか否かを判断する（ステップ３０２）。ＴＨＥＮ文が存在する場合は、ステップ３０１～ステップ３１１の処理を再帰的に呼び出す（ステップ３０３）。一方、ＴＨＥＮ文が存在しない場合は、ソースプログラム上のその位置にＴＨＥＮ文及びダミー処理を挿入し（ステップ３０４）、チェックポイントを設定する（ステップ３０５）とともに、条件及び処理の情報を条件分岐テーブル１０８に登録する（ステップ３０６）。

【００３６】次に、現在解析を行っている条件分岐命令（必ずしも最上位の条件分岐命令とは限らず、再帰的に呼び出されている場合は、下位の条件分岐命令である可能性もある。）の下にＥＬＳＥ文が存在するか否かを判断する（ステップ３０７）。ＥＬＳＥ文が存在する場合は、ステップ３０１～ステップ３１１の処理を再帰的に呼び出す（ステップ３０８）。一方、ＥＬＳＥ文が存在しない場合は、ソースプログラム上のその位置にＥＬＳＥ文及びダミー処理を挿入し（ステップ３０９）、チェックポイントを設定する（ステップ３１０）とともに、条件及び処理の情報を条件分岐テーブル１０８に登録する（ステップ３１１）。

【００３７】また、ステップ３０１で条件分岐命令でないと判断された場合には、読み込んだ処理の位置にチェックポイントを設定する（ステップ３１２）とともに、条件及び処理の情報を条件分岐テーブル１０８に登録する（ステップ３１３）。

【００３８】このようにして、分解済ソースプログラム１０３が作成される。

【００３９】図２を参照すると、分解済ソースプログラム１０３においては、条件Ａが真かつ条件Ｂが偽かつ条件Ｃが真の位置にダミー処理Ｘが挿入され、条件Ａが偽の位置にダミー処理Ｙが挿入されている。さらに、処理Ｐに対してチェックポイント１が、ダミー処理Ｘに対してチェックポイント２が、処理Ｑに対してチェックポイント３が、ダミー処理Ｙに対してチェックポイント４が設定されている。

【００４０】ここで、ダミー処理とは、処理内容を伴わない処理であり、実際には何も実行されず、条件をＢＲＥＡＫするだけのものである。

【００４１】また、図２を参照すると、条件分岐テーブル１０８は、初期ソースプログラム１０１に含まれる条件Ａ、Ｂ、Ｃの真偽と、その時に行う処理との対応情報を格納する。条件分岐テーブル中の数字「１」は、対応する条件が真の時に対応する処理を実行することを示し、数字「０」は、対応する条件が偽の時に対応する処理を実行することを示す。また、「ＮＵＬＬ」は、対応する条件を判断せずに対応する処理を実行することを示している。したがって、例えば、条件Ａが真かつ条件Ｂが真の時に処理Ｐを実行することが分かる。

【００４２】なお、本実施例の条件分岐テーブルでは、条件がＡ、Ｂ、Ｃの３つであるが、条件の数は必要に応じて設定できる。

【００４３】また、図１を参照すると、コンパイル手段１０４は、分解済ソースプログラム１０３をコンパイルして、試験用ロードモジュール１０５を作成する。そして、統計データ蓄積手段１０７は、商用標準データ１０６を使用して試験用ロードモジュール１０５を実行することにより、条件分岐テーブル１０８を更新する。統計データ蓄積手段１０７の動作について、図４～図５を参照して説明する。

【００４４】図４に、試験用ロードモジュール１０５、商用標準データ１０６、条件分岐テーブル１０８の一例を示す。なお、試験用ロードモジュール１０５については、便宜上、ソースプログラムのイメージで示している。

【００４５】この例では、試験用ロードモジュール１０５は、条件Ａが真かつ条件Ｂが真の時に処理Ｐを実行し、条件Ａが真かつ条件Ｂが偽かつ条件Ｃが偽の時に処理Ｑを実行し、条件Ａが真かつ条件Ｂが偽かつ条件Ｃが真の時にダミー処理Ｘを実行し、条件Ａが偽の時にダミー処理Ｙを実行するように組まれている。

【００４６】また、商用標準データ１０６は、各条件に対して図に示すような分岐を行うデータから構成されている。例えば、１件目のデータは、条件Ａが真かつ条件Ｂが偽かつ条件Ｃが真となるようなデータである。また、２件目のデータは、条件Ａが偽となるようなデータである。

【００４７】図５は、統計データ蓄積手段１０７が試験用ロードモジュール１０５と商用標準データ１０６に基づいて条件分岐テーブル１０８を更新する際の動作をフローチャートにより示したものである。

【００４８】まず、統計データ蓄積手段１０７は、条件分岐テーブル１０８における件数の初期化を行う（ステップ５０１）。

【００４９】そして、商用標準データ１０６を１件読み込み（ステップ５０２）、条件分岐テーブル１０８において、この時通過したチェックポイントに対応する件数に１を加算する（ステップ５０３）。

【００５０】さらに、統計データ蓄積手段１０７は、未処理のデータが他にあるか否かを判断し（ステップ５０４）、ある場合には、ステップ５０２～ステップ５０３の処理を繰り返す。

【００５１】このようにして、条件分岐テーブル１０８

の更新が行われる。

【００５２】図４を参照すると、条件分岐テーブル１０８は、処理がチェックポイント１を通過した件数が２件、チェックポイント２を通過した件数が８件、チェックポイント３を通過した件数が１０件、チェックポイント４を通過した件数が５件であることを示している。すなわち、処理Ｐによって処理される件数が２件、ダミー処理Ｘによって処理される件数が８件、処理Ｑによって処理される件数が１０件、ダミー処理Ｙによって処理される件数が５件であることを示している。

【００５３】さらに、図１を参照すると、条件分岐最適化手段１０９は、分解済ソースプログラム１０３を読み込み、条件分岐テーブル１０８に基づいて最適化ソースプログラム１１０を作成する。条件分岐最適化手段１０９の動作について、図６～図７を参照して説明する。

【００５４】図６に、分解済ソースプログラム１０３、条件分岐テーブル１０８、最適化ソースプログラム１１０の一例を示す。

【００５５】この例では、分解済ソースプログラム１０３においては、条件Ａが真かつ条件Ｂが偽かつ条件Ｃが真の位置にダミー処理Ｘが挿入され、条件Ａが偽の位置にダミー処理Ｙが挿入されている。さらに、処理Ｐに対してチェックポイント１が、ダミー処理Ｘに対してチェックポイント２が、処理Ｑに対してチェックポイント３が、ダミー処理Ｙに対してチェックポイント４が設定されている。

【００５６】また、条件分岐テーブル１０８は、処理がチェックポイント１を通過した件数が２件、チェックポイント２を通過した件数が８件、チェックポイント３を通過した件数が１０件、チェックポイント４を通過した件数が５件であることを示している。すなわち、処理Ｐによって処理される件数が２件、ダミー処理Ｘによって処理される件数が８件、処理Ｑによって処理される件数が１０件、ダミー処理Ｙによって処理される件数が５件であることを示している。

【００５７】図７は、条件分岐最適化手段１０９が分解済ソースプログラム１０３と条件分岐テーブルに基づいて最適化ソースプログラム１１０を作成する際の動作をフローチャートにより示したものである。

【００５８】条件分岐最適化手段１０９は、条件分岐テーブル１０８を件数の大きい順に並べ替える（ステップ７０１）。

【００５９】図６に示すような条件分岐テーブル１０８に対しては、チェックポイント３を通過する処理Ｑの件数が１番多いので、処理Ｑが１番目になるようにし、次いで、ダミー処理Ｘ、ダミー処理Ｙ、処理Ｐの順になるように並べ替える。

【００６０】また、条件分岐最適化手段１０９は、ＩＦ～ＴＨＥＮ～ＥＬＳＥによって記述されたソースプログラムの形式をＣＡＳＥ～ＯＦに変換する（ステップ７０２）。

【００６１】次に、条件分岐テーブル１０８から、１つの処理に相当するレコードを読み込む（ステップ７０３）。さらに、その処理を実行すべき条件を記述した後（ステップ７０４）、その処理を記述する（ステップ７０５）。

【００６２】さらに、条件分岐最適化手段１０９は、未処理の条件節が他にあるか否かを判断し（ステップ７０６）、ある場合には、ステップ７０１～ステップ７０５の処理を繰り返す。

【００６３】このようにして、最適化ソースプログラム１１０が作成される。

【００６４】図６を参照すると、最適化ソースプログラム１１０は、条件Ａが真かつ条件Ｂが偽かつ条件Ｃが偽の時に処理Ｑを実行し、条件Ａが真かつ条件Ｂが偽かつ条件Ｃが真の時にダミー処理Ｘを実行し、条件Ａが偽の時にダミー処理Ｙを実行し、条件Ａが真かつ条件Ｂが真の時に処理Ｐを実行するように組まれている。

【００６５】図１を参照すると、最後に、コンパイル手段１０４は、最適化ソースプログラム１１０をコンパイルして、運用ロードモジュール１１１を作成する。

【００６６】以上により、本発明の一実施例であるソースプログラムの最適化装置、あるいは最適化方法を適用したシステムの処理が終了する。

【００６７】なお、本実施例においては、条件分岐命令として、ＩＦ文、ＴＨＥＮ文、ＥＬＳＥ文を含むものを考えたが、必ずしもこれらの文である必要はなく、本発明は、ＩＦ文、ＴＨＥＮ文、ＥＬＳＥ文の機能に相当する他の文についても適用することができる。また、ソースプログラム中のＩＦ文等の二者択一型の条件分岐文だけでなく、ＣＡＳＥ文等の多肢選択型の条件分岐文に対しても、同様にして、本発明を適用することが可能である。

【００６８】本発明の一実施例であるソースプログラムの最適化装置、あるいは最適化方法を適用したシステムは、複雑な条件分岐を多岐条件分岐に変換するとともに、条件節をその成立頻度の順に並べ変えることにより、プログラムを高速に実行できるようにすることができるという効果を有している。

【００６９】また、複雑な条件分岐を多岐条件分岐に変換し、条件節をその成立頻度の順に並べ変えた後にコンパイルを行うことにより、高速に実行可能に変更されたソースプログラムを提供することができるという効果を有している。

【００７０】

【発明の効果】以上説明したように、本発明のソースプログラムの最適化装置および最適化方法は、プログラムを高速に実行できるようにすることができるという効果を有している。

【００７１】また、高速に実行可能に変更されたソース

プログラムを提供することができるという効果を有している。

【図面の簡単な説明】

【図１】図１は本発明の一実施例の構成を示すブロック図である。

【図２】図２は本発明の一実施例における条件分岐分解手段１０２の動作の概要を示す図である。

【図３】図３は本発明の一実施例における条件分岐分解手段１０２の処理を示す流れ図である。

【図４】図４は本発明の一実施例における統計データ蓄積手段１０７の動作の概要を示す図である。

【図５】図５は本発明の一実施例における統計データ蓄積手段１０７の処理を示す流れ図である。

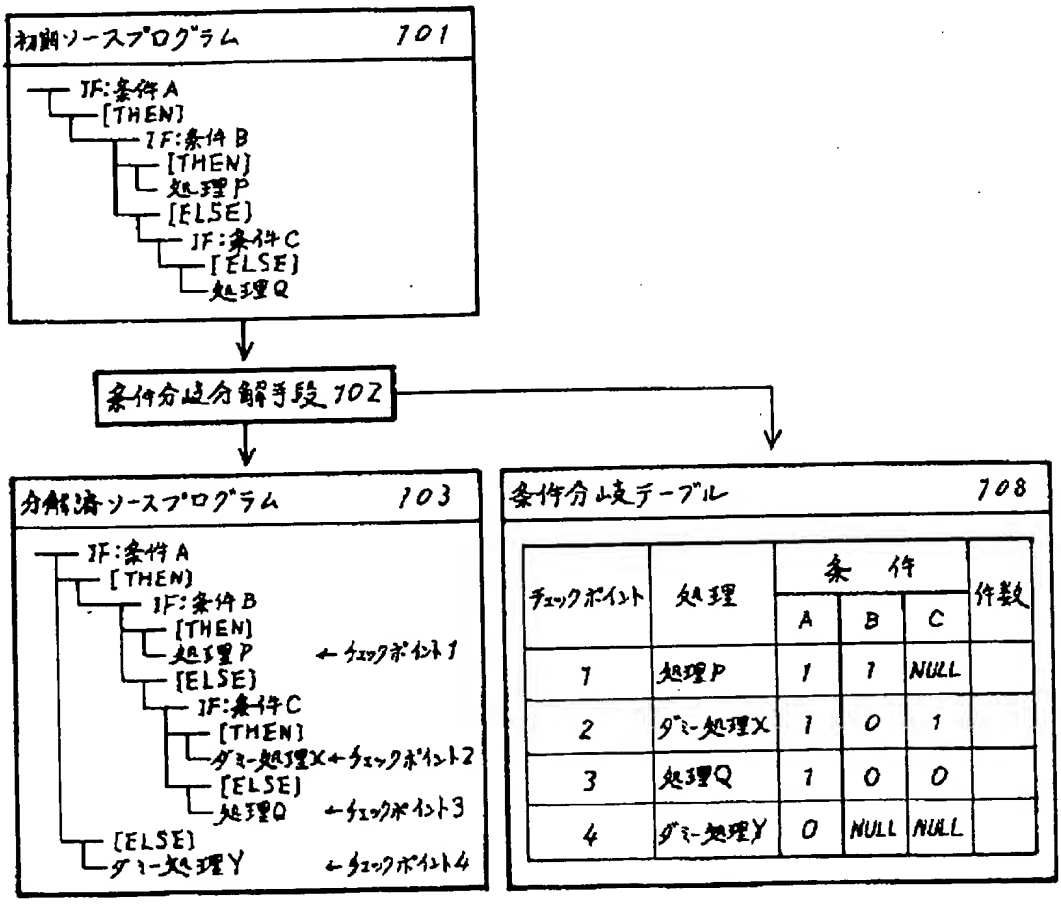
【図６】図６は本発明の一実施例における条件分岐最適化手段１０９の動作の概要を示す図である。

【図７】図７は本発明の一実施例における条件分岐最適化手段１０９の処理を示す流れ図である。

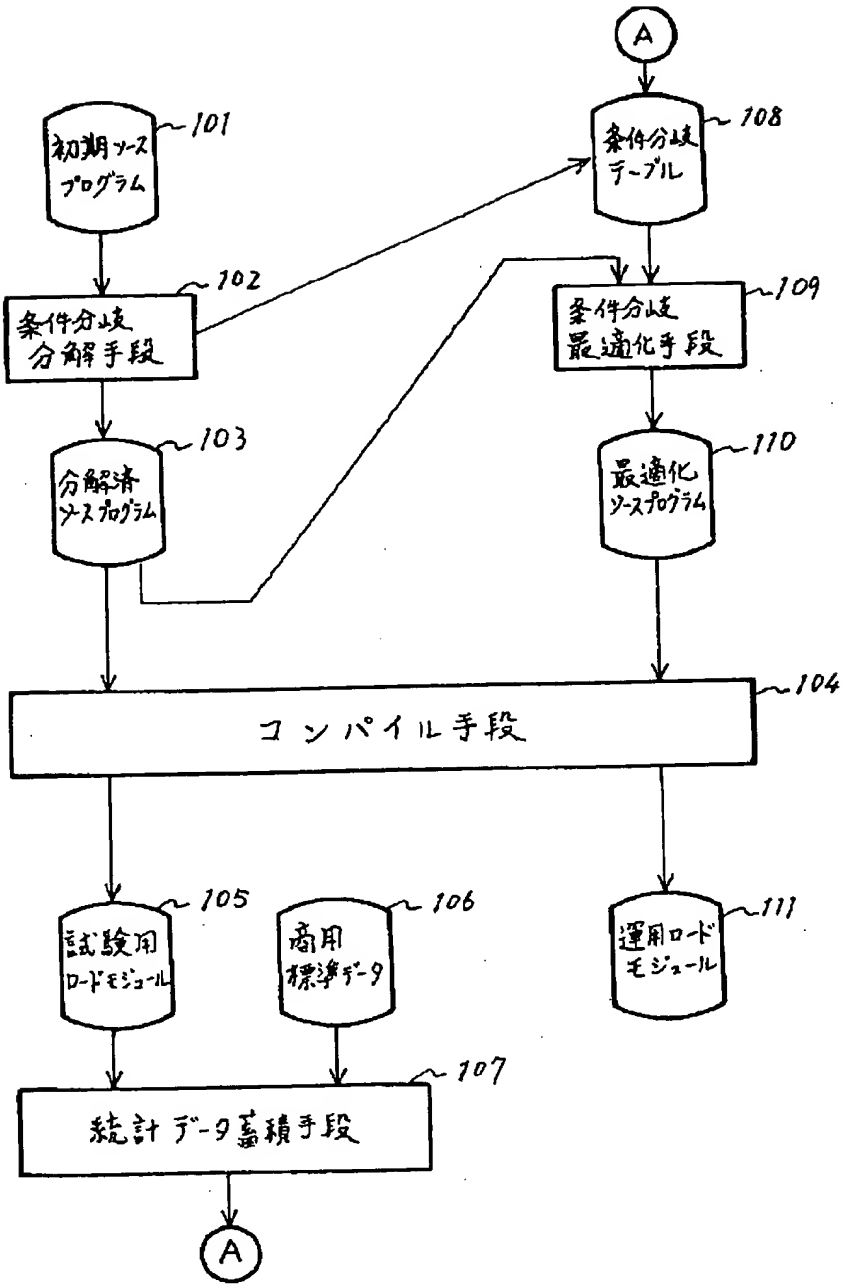
【符号の説明】

１０１　初期ソースプログラム
１０２　条件分岐分解手段
１０３　分解済ソースプログラム
１０４　コンパイル手段
１０５　試験用ロードモジュール
１０６　商用標準データ
１０７　統計データ蓄積手段
１０８　条件分岐テーブル
１０９　条件分岐最適化手段
１１０　最適化ソースプログラム
１１１　運用ロードモジュール

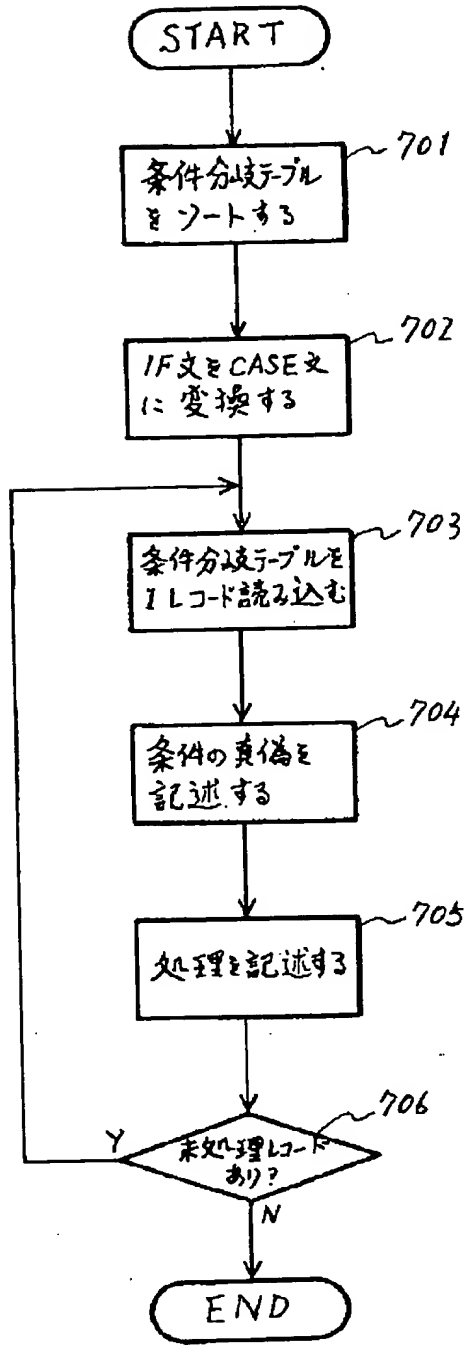
【図２】



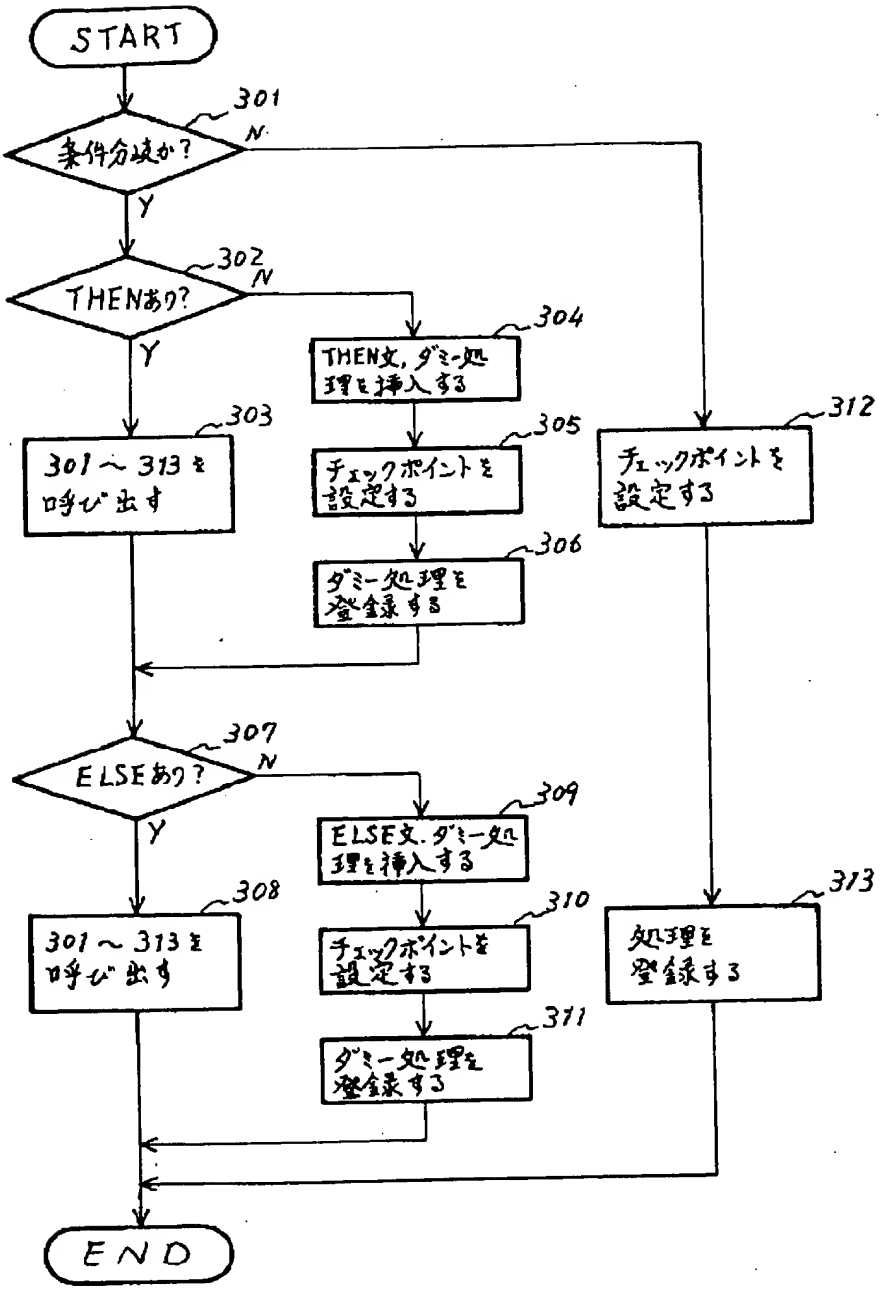
【図１】


A
初期ソースプログラム
101
条件分岐テーブル
108
条件分岐分解手段
102
条件分岐最適化手段
109
分解済ソースプログラム
103
最適化ソースプログラム
110
コンパイル手段
104
試験用ロードモジュール
105
商用標準データ
106
運用ロードモジュール
111
統計データ蓄積手段
107
A

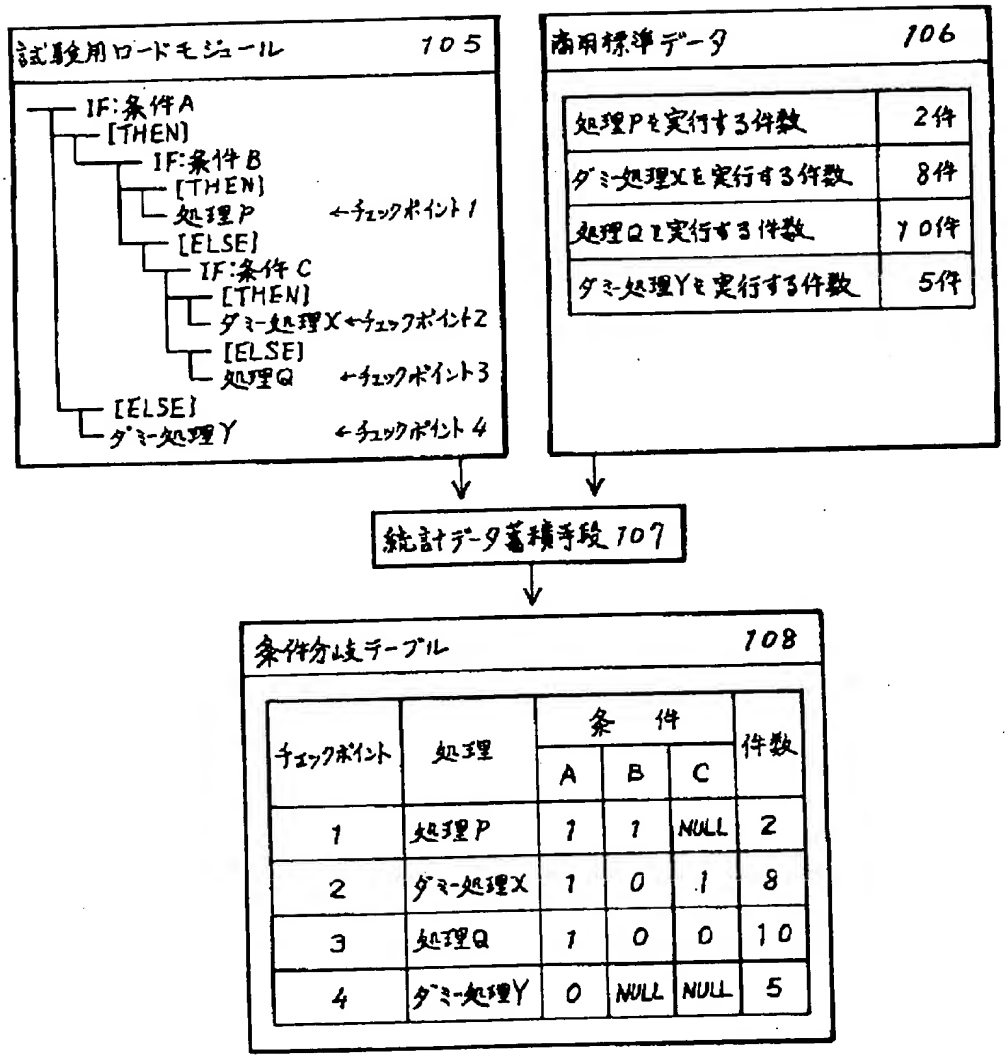
【図７】


START
条件分岐テーブルをソートする
701
IF文をCASE文に変換する
702
条件分岐テーブルを1Lコード読み込む
703
条件の真偽を記述する
704
処理を記述する
705
未処理レコードあり?
706
Y
N
END

【図３】

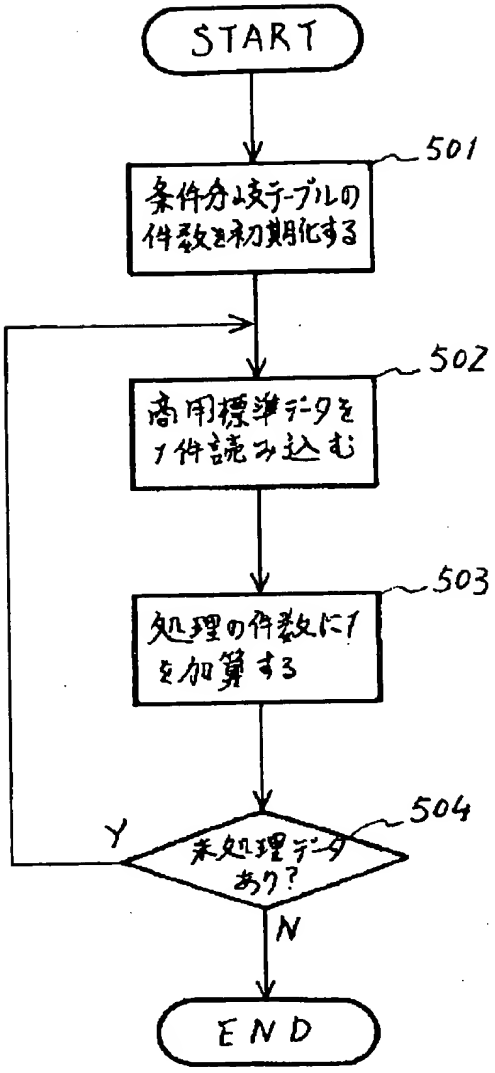

START
301
条件分岐か?
N
Y
302
THENあり?
N
Y
304
THEN文,ダミー処理を挿入する
303
301～313を呼び出す
305
チェックポイントを設定する
306
ダミー処理を登録する
312
チェックポイントを設定する
307
ELSEあり?
N
Y
309
ELSE文,ダミー処理を挿入する
308
301～313を呼び出す
310
チェックポイントを設定する
311
ダミー処理を登録する
313
処理を登録する
END

【図4】

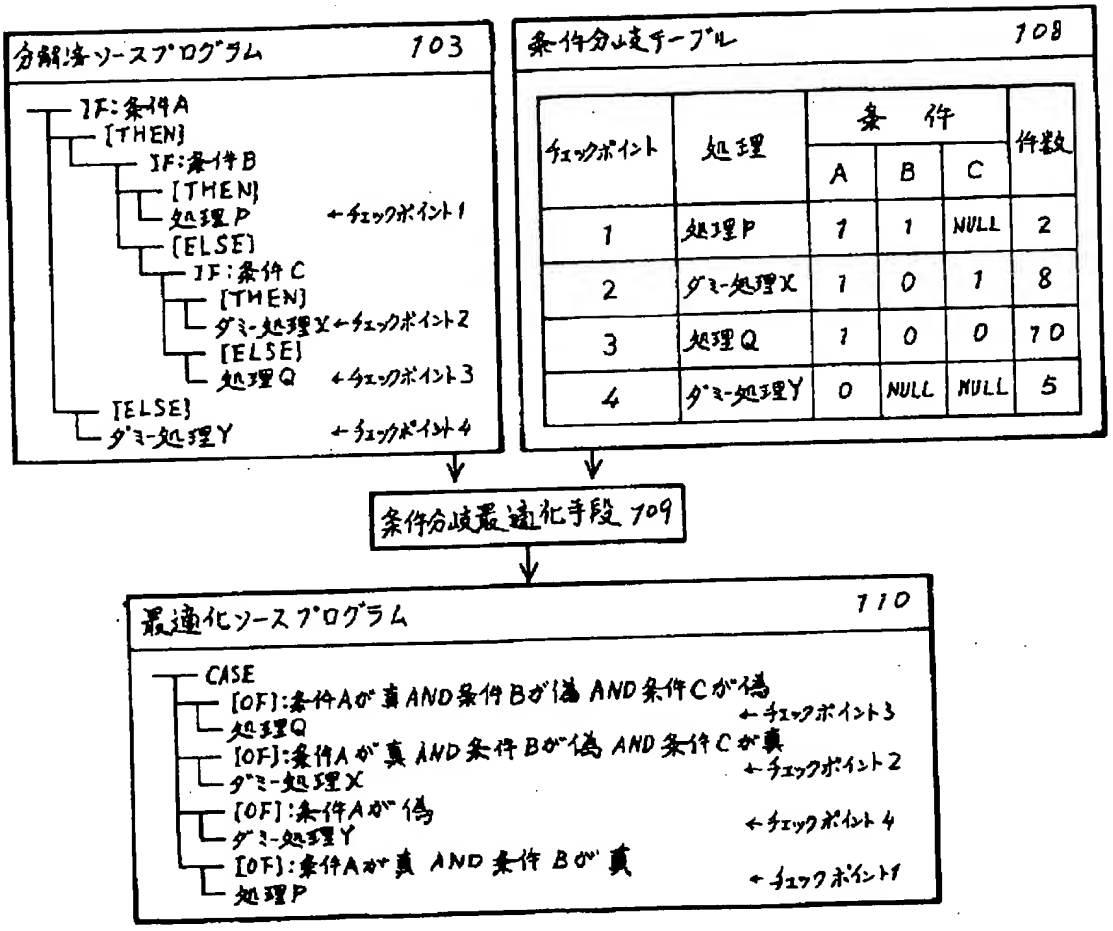


チェックポイント	処理	条件			件数
		A	B	C	
1	処理P	1	1	NULL	2
2	ダミー処理X	1	0	1	8
3	処理Q	1	0	0	10
4	ダミー処理Y	0	NULL	NULL	5

【図５】


START
501
条件分岐テーブルの件数を初期化する
502
商用標準データを1件読み込む
503
処理の件数に1を加算する
504
未処理データあり?
Y
N
END

【図6】


分解済ソースプログラム 103
IF:条件A
[THEN]
IF:条件B
[THEN]
処理P ←チェックポイント1
[ELSE]
IF:条件C
[THEN]
ダミー処理X ←チェックポイント2
[ELSE]
処理Q ←チェックポイント3
[ELSE]
ダミー処理Y ←チェックポイント4
条件分岐テーブル 108
チェックポイント
処理
条件
A
B
C
件数
1 処理P 1 1 NULL 2
2 ダミー処理X 1 0 1 8
3 処理Q 1 0 0 10
4 ダミー処理Y 0 NULL NULL 5
条件分岐最適化手段 109
最適化ソースプログラム 110
CASE
[OF]:条件Aが真AND条件Bが偽AND条件Cが偽 ←チェックポイント3
処理Q
[OF]:条件Aが真AND条件Bが偽AND条件Cが真 ←チェックポイント2
ダミー処理X
[OF]:条件Aが偽 ←チェックポイント4
ダミー処理Y
[OF]:条件Aが真 AND 条件Bが真 ←チェックポイント1
処理P